# Korespondenční Seminář z Programování

## ZAČÁTEČNICKÁ KATEGORIE

**35. ročník** **KSP-Z** **Únor 2022**

Právě se díváte na leták čtvrté série 35. ročníku **KSP-Z**, neboli **K**orespondenčního **S**emináře z **P**rogramování, **Z**ačátečnické kategorie. Zapojit se může **každý středoškolák i základoškolák**, řešit můžete začít i v případě, kdy jste se nezúčastnili prvních sérií. Ty nejúspěšnější z vás na jaře pozveme na **týdenní soustředění** (pokud to aktuální situace dovolí), na kterém se toho spoustu dozvíte a zároveň si užijete hromadu neopakovatelné zábavy.

Letos bude v KSP-Z **pět sérií** po **čtyřech úlohách** za celkem **220 bodů**. Pokud budete mít jakoukoliv otázku, neváhejte se zeptat. Kontaktní adresy najdete v patičce na konci letáku. Přejeme hodně štěstí!

**Termín série:** **neděle 2. dubna ve 32:00** (tedy další ráno v 8:00), praktické úlohy za třetinu bodů až do 9. dubna
**Obsah série:** **3 praktické úlohy** (značené 🖩) – K těmto úlohám je nutné napsat program (v libovolném vhodném jazyce), stáhnout si z našeho webu vstupní data a odevzdat odpovídající výstup. **1 teoretická úloha** (značená 💡) – U této úlohy nás zajímá hlavně slovní popis řešení, ve kterém byste měli zdůvodnit jeho funkčnost a ideálně nás i přesvědčit o jeho efektivitě.
**Odevzdávání:** Přes web na adrese *https://ksp.mff.cuni.cz/z/odevzdavatko/*

## Zadání čtvrté série začátečnické kategorie 35. ročníku KSP

### 🖩 35-Z4-1 Lampy v ulici — 8 bodů

I do města Hrochovníku dorazil nedostatek energií. Rada starších hrochů se proto rozhodla začít šetřit na pouličním osvětlení.

V Hrochovníku je pouze jedna osvětlená hlavní ulice, podél které jsou v pravidelných rozestupech umístěné lampy. Hrochové rozhodli, že je možné vypnout lampy tak, aby každá vypnutá lampa byla obklopena dvojicí svítících lamp. Některé lampy jsou již teď rozbité a tedy nesvítí (a na jejich opravu nejsou peníze).

Navrhněte, které lampy nechat svítit tak, aby jejich provoz byl co nejúspornější, tedy svítilo co nejméně lamp.

Pokud existuje více stejně dobrých řešení, vypište libovolné z nich. Máte zaručeno, že alespoň jedno řešení existuje, tedy všechny rozbité lampy jsou obklopeny svítícími lampami.

Toto je praktická open-data úloha. V odevzdávacím systému si necháte vygenerovat vstupy a odevzdáte příslušné výstupy. Záleží jen na vás, jak výstupy vyrobíte.

*Formát vstupu:* Na vstupu dostanete jeden řetězec popisující situaci v Hrochovníku. Každý znak reprezentuje jednu lampu. Když je to `O`, tak lampa je v pořádku, ovšem když je to `X`, tak je rozbitá.

*Formát výstupu:* Vypište jedno z optimálních řešení, kde svítí co nejméně lamp, přitom každá nesvítící lampa je obklopena dvojicí svítících. Řešení vypište na jeden řádek, pro každou lampu jeden z následujících znaků:

- `O` – svítící lampa
- `-` – zhasnutá lampa
- `X` – rozbitá lampa

| *Ukázkový vstup:* | *Ukázkový výstup:* |
|---|---|
| `OOOXOXOOOOOO` | `O-OXOXO-O-OO` |

### 🖩 35-Z4-2 Nudná hodina — 10 bodů

Matematiku dnes supluje dějepisář Kaiser. Plete páté přes deváté, tak Sára s Petrem zahánějí nudu, jak se dá. Už vyřešili všechny úlohy z Prasátka, přečetli celé Matematicko-fyzikální tabulky a teď soutěží v rozkládání čísel na součin prvočísel. Kevin je pozoruje a trochu závidí, jak rychle jim to jde. Aby mohl doma trénovat, chce si napsat program, který bude prvočíselné rozklady počítat. Zkuste to také.

Například číslo 2024 je rovno $2 \cdot 2 \cdot 2 \cdot 11 \cdot 23$, což můžeme zkráceně zapsat jako $2^3 \cdot 11 \cdot 23$.

Toto je praktická open-data úloha. V odevzdávacím systému si necháte vygenerovat vstupy a odevzdáte příslušné výstupy. Záleží jen na vás, jak výstupy vyrobíte.

*Formát vstupu:* Na prvním řádku dostanete $N$ – počet čísel, která chce Kevin rozložit. Na každém z následujících $N$ řádků se bude nacházet jedno přirozené číslo v rozsahu od 2 do 1 000 000 000.

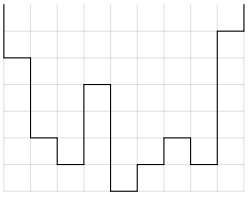
*Formát výstupu:* Pro každé z $N$ zadaných čísel vypište jeden řádek s rozkladem na součin prvočísel. Jednotlivá prvočísla vypisujte v rostoucím pořadí a oddělujte znakem `*`. Pokud se nějaké prvočíslo vyskytne vícekrát, zapište ho jako mocninu ve tvaru `p^k` (znak „stříška“). Mezi čísly a znaménky nevypisujte žádné mezery.

| *Ukázkový vstup:* | *Ukázkový výstup:* |
|---|---|
| `3` | `7*17^2` |
| `2023` | `2^3*11*23` |
| `2024` | `2027` |
| `2027` | |

### 🖩 35-Z4-3 Záchrana kachničky — 12 bodů

Kristýna si pouštěla svoji gumovou kachničku na rybníce. Bohužel se ale rybník začal vypouštět a kachnička zůstala uvízlá uprostřed rybníka. Kristýna samozřejmě nechce do rybníka vlézt, protože by se ušpinila. A tak vymyslela, že bude do rybníka lít vodu, dokud kachnička nevyplave.

Dno rybníka si můžeme představit ve čtvercové mřížce jako sloupce s různou výškou. Břehy rybníka tvoří nekonečné sloupce, přes které voda nikdy nepřeteče. Například rybník s výškami sloupců $5, 2, 1, 4, 0, 1, 2, 1, 6$ by vypadal takto:



Kristýna lije vodu do rybníka z levého břehu, kde je pumpa na vodu. Dokud dno rybníka klesá, voda stéká dolů. Jakmile doteče do místa, kde se dno zvedá, začne hladina vody stoupat. Pokud vystoupá dostatečně vysoko, začne stejným způsobem přetékat do dalšího příkopu.

Kachnička se nachází na jednom ze sloupců a vyplave, když pod ní bude 1 jednotka vody. (Můžeme si představit, že voda ji ponese ke břehu ve třetí ose kolmé na obrázek.)

Řekněte Kristýně, kolik bude zapotřebí vody, než kachnička vyplave.

Toto je praktická open-data úloha. V odevzdávacím systému si necháte vygenerovat vstupy a odevzdáte příslušné výstupy. Záleží jen na vás, jak výstupy vyrobíte.

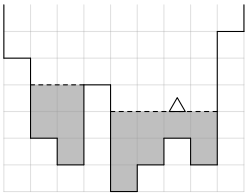
*Formát vstupu:* Na prvním řádku dostanete dvě čísla $N$ a $D$ – počet sloupců a pozici kachničky (číslujeme od jedné). Na druhém řádku dostanete $N$ čísel – výšky jednotlivých sloupců.

*Formát výstupu:* Vypište jedno číslo – kolik vody je zapotřebí, aby kachnička vyplavala.

Pozor, výsledek může být opravdu velký, proto použijte 64bitovou proměnnou (`long long` v C++). V Pythonu toto řešit nemusíte.

| *Ukázkový vstup:* | *Ukázkový výstup:* |
|---|---|
| `9 7`<br>`5 2 1 4 0 1 2 1 6` | `13` |

Jak bude rybník vypadat po 13 jednotkách vody můžete vidět na obrázku:



## 35-Z4-4 Těžba zlata — 14 bodů

Kevina popadla zlatá horečka! Došlo mu, že těžení zlata je přeci nejlepší způsob, jak vydělávat.

Kevin ale nepatří mezi fanatiky, co hned začnou kopat do země a ani neví kam. Rozhodl se, že vše dobře naplánuje.

Nejprve vykopal vstupní šachtu do hloubky, kde se zlato nachází, a posledních několik týdnů dělal detailní průzkum jejího okolí. Ví tedy, kde přesně ze nachází zlaté žíly.

Okolí vstupní šachty si můžeme přestavit jako 2D mřížku. Na jednom z políček se nachází vstupní šachta. Zbylá políčka jsou buď kámen, nebo zlato.

Nyní by Kevin konečně rád začal těžit. Ovšem po tolika týdnech práce už je unavený. Najal si tedy levnou pracovní sílu (tedy Zuzku), která bude kopat za něj.

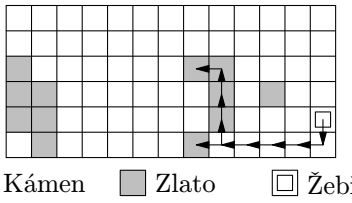
Zuzka je ochotná za jeden Krtkoin vykopat libovolné políčko, které hranou sousedí s šachtou nebo s již vykopaným políčkem. Kevin dále ví, že prodejem zlata z jednoho políčka získá deset Krtkoinů.

Těžení má ovšem ještě jeden háček: jakmile Zuzka uvidí nějaké políčko se zlatem (políčko bude hranou sousedit s nějakým již vytěženým políčkem), již nikdy nebude chtít kopat kámen. Tedy již bude ochotná těžit jen zlato, pak jí Kevin zaplatí za celou práci a půjde domů.

Navrhněte, v jakém pořadí má Kevin po Zuzce chtít, aby těžila políčka. Kevin by si samozřejmě rád vydělal co nejvíce. Pokud je optimálních postupů více, stačí určit pouze jeden z nich.

Toto je teoretická úloha. Není nutné ji programovat, odevzdává se pouze slovní popis algoritmu. Více informací zde: `http://ksp.mff.cuni.cz/viz/tinfo`

*Příklad:*




Kevin musí navést Zuzku tak, aby minula malé ložisko zlata poblíž vstupu. Pak by totiž vytěžila jenom jej a šla domů. Také tím, že ji navede do políčka, odkud jsou dostupná dvě ložiska, získá více zlata. Největší ložisko je už moc daleko, takže se k němu nevyplatí chodit.

## Praktický kurz programování

Pokud Tě lákají praktické úlohy, ale ještě neumíš žádný programovací jazyk, můžeš se podívat na náš Základní kurz programování, kde se můžeš naučit základy Pythonu: *https://ksp.mff.cuni.cz/kurz/* .

## Zdrojáky praktických úloh

Řešení praktických úloh může být ze začátku složité. Velmi často i nějaká triviální technická chyba ve zdrojovém kódu programu může znamenat, že program vrací špatný výsledek – a některé chyby se ze začátku špatně hledají. Proto Ti nabízíme možnost poslat zdrojový kód programu nějaké úlohy na adresu *zdrojaky@ksp.mff.cuni.cz*, kde se Ti pokusíme poradit. Do emailu prosím připiš:

- Jakou úlohu by měl program řešit.
- Slovní popis, co by měl program podle Tebe dělat.

Před termínem série Ti nemůžeme radit s algoritmem, ale pomůžeme s odladěním zdrojáku. Po termínu série pak můžeme poradit i s návrhem algoritmu – získáš tak znalosti do dalších sérií.

matfyz

KSP pro vás připravují studenti Matematicko-fyzikální fakulty Univerzity Karlovy.
Realizace projektu byla podpořena Ministerstvem školství, mládeže a tělovýchovy.

**Webové stránky:** *https://ksp.mff.cuni.cz/*
**E-mail:** *ksp@mff.cuni.cz*
**Organizátoři a kontakty:** *https://ksp.mff.cuni.cz/kontakty/*